July/August 1986 Edition

RE RUN

RUN Programs on Disk

*For the C-64 and C-128**

Plus: Bonus Programs!

*128 mode programs included

Introduction

July/August '86 ReRUN

Welcome to our July/August edition of ReRUN. This disk holds a variety of useful programs for both programmers and application users. (In the descriptions below, article titles are in parentheses if different from the disk filenames.)

RUN's July issue focused on graphics, offering three applications and utilities for C-64 and C-128 owners. HI-RES WRITER is a 64 utility that lets you place text on your high resolution screen displays.

NEEDLEGRAPH is a 64 application designed to please craftsmen who rely on patterns and color for their creations. The program lets you design your own variegated squares, save them and print them out for future use.

C-128 owners can get a peek at some of the nifty tricks their computer can perform in 80-columns from both C-64 and C-128 modes. Try our four short demo programs, VIDEO 1, 2, 3 and 4 (Video Fantasia).

RUN continues to provide its readers with useful applications that will make life easier. Our July issue held a noteworthy application (Extra! Extra! Read All About It) for desktop publishers. It tells you how your group can save money and time by creating disk-based newsletters on your C-64. Three programs, MAG PRODUCTION, MAG READER and MAG PRINTER, are the core needed to create, edit, read and print out your newsletter. In order to get going with the production side, you will, of course, need a word processor. RUN Script 64 is suggested.

Our August issue held a handy autoboot program for C-128 owners who'd like the luxury of utilizing the new computer's autoboot feature for their C-64 disks. BOOTMAKER 128/64 lets you do just that by simply performing a few preparatory steps.

C-64 owners will find DISK KEEPER an invaluable housekeeping tool. This program lets you do everything from soup to nuts with your disks. Need a directory? Format a disk? Rename a disk? Protect your files from being accidentally deleted? DISK KEEPER does all these things and makes printouts of your di-

rectories as well. If that's not enough, it will also print out your disk jackets.

MAKECHARS and USECHARS (Add Some Character to Your C-64) round out our August offerings. These two programs take advantage of the C-64's unique ability to redefine any character on the keyboard. They also give you the flexibility to switch placement of keys (for example, to create a DVORAK keyboard), or to develop a new character set (the German alphabet, for instance).

As always, we have bonus programs for you. This edition contains a super three-dimensional graphics editor for C-128 mode users. 3D OBJECT EDITOR is the first serious mathematical application that I have seen for the C-128. Included are three demonstration programs.

C-64 owners have MENU SUBROUTINE, a short routine that can be merged into your own programs to make menu programming easier and your programs more user-friendly.

Margaret Morabito
Technical Manager
RUN *Magazine*

Directory

	MENU 128	C-128 MODE	BASIC
	MENU 64	C-64	BASIC
1	VIDEO 1	C-128 MODE	BASIC
1	VIDEO 2	C-128 MODE	BASIC
1	VIDEO 3	C-128 MODE	BASIC
1	VIDEO 4	C-128 MODE	BASIC
5	HI-RES WRITER	C-64	BASIC
9	NEEDLEGRAPH	C-64	BASIC
13	MAG PRODUCTION	C-64	BASIC
13	MAG READER	C-64	BASIC
13	MAG PRINTER	C-64	BASIC
21	BOOTMAKER 128/64	C-128 MODE	BASIC
23	DISK KEEPER	C-64	BASIC
28	MAKECHARS	C-64	BASIC
28	USECHARS	C-64	BASIC
£32	MENU SUBROUTINE	C-64	BASIC
£36	3D OBJECT EDITOR	C-128 MODE	BASIC (40 columns)
+36	3D DEMO.1	C-128 MODE	SEQ
+36	3D DEMO.2	C-128 MODE	SEQ
+36	3D DEMO.3	C-128 MODE	SEQ

£ Bonus program!

\+ These are sequential files, which can be loaded only from within 3D OBJECT EDITOR.

How To Load

This ReRUN provides menu programs for both C-64 and C-128 users. C-64 users should type LOAD "MENU 64" and RUN. C-128 users need only press the shift and run/stop keys. These menus will display all of the programs on the disk and let you run them with a single key press.

If you don't want to use the menu programs, then follow these guidelines for loading individual programs.

C-64:

To load a C-64 program written in Basic, type:

LOAD "PROGRAM NAME",8

and then press the return key. The drive will whir while the screen prints LOADING, and then READY, with a flashing cursor beneath. Type RUN and press the return key. The program will then begin.

Machine language (ML) programs will be loaded with LOAD "PROGRAM NAME",8,1. When these occur, they will be clearly marked for you on the directory page.

C-128 in C-64 mode:

All C-64 programs can be run on the C-128 as long as your computer is in C-64 mode. This is accomplished by typing GO64 after powering up your C-128. Answer Y and press the return key in response to the "Are you sure?" prompt. When in C-64 mode, use the instructions above for loading C-64 programs.

C-128 in C-128 mode:

All C-128 mode programs are clearly labeled on the directory page. Your C-128 must be in C-128 mode to run these programs. All programs will work in both 40 and 80 columns, unless they are specifically marked otherwise.

To load a C-128 mode program, just press F2 and type the name of the program that you wish to load. Use the exact name that appears on the directory page. When the program has loaded, a READY prompt will appear. Beneath this is the flashing cursor where you should type RUN and press the return key. The program will then run.

NOTES:

Before loading any program, always refer to the article in this booklet for special instructions.

It is a good idea to make a copy of your ReRUN disk before running the programs. You can accomplish this by using the disk backup programs that Commodore provides for the 1541 and 1571 disk drives.

ReRUN Staff

Technical Manager/Editor: *Margaret Morabito*
Technical Editor: *Tim Walsh*
Art Director: *Glenn A. Suokko*
Design and Layout: *Karla M. Whitney*
Typesetting: *Doreen Means*

Video Fantasia

By Noel Nyman and Chris Newman

RUN It Right

C-128 (in 80-column mode)

One of the best features of the C-128 is its ability to switch between 40- and 80-column modes. But did you know that you can run programs in C-64 mode and watch their execution in 80-column 128 mode? This article provides the programs and know-how needed to perform these remarkable feats.

To understand how the 80-column screen works, it helps to think of it as a peripheral, not unlike a printer or disk drive, which is sent information one bit or byte at a time through a "port." The computer gives the peripheral some data, along with an instruction on how to use it. The device then manipulates the data, using its own processor and memory.

The C-128's 8563 video controller generates the 80-column screen and uses it like a peripheral. The 8563 has 16K of RAM, which holds the character sets and the screen and color memories. The controller receives and sends data and instructions via two address locations (ports) in the C-128 to make changes in the screen display. Once you know how to control the 8563, you can make any changes you like on the 80-column screen, even from C-64 mode!

To see how the 8563 controls screen functions, first start up your C-128 in 80-column mode. Next, type the following command in Direct mode and press the return key.

```
POKE 54784,24: POKE 54785,
XOR(PEEK(54785),64)
```

You should see the entire 80-column screen reverse. If that's not what happens, check to be sure you typed in the command correctly.

The command first Pokes a 24 at 54784 to tell the 8563 that you want to work with register 24. The second part of the command first Peeks location 54785 to look at the value from register 24, and then, using XOR, it combines the value 64 with the data from location 54784 and puts the result back in the same location.

Each time you execute this command, the screen will toggle to the opposite video mode.

Let's try a different register with some underlined text on the screen. Press and release the CTRL and B keys simultaneously; then type a few words. They should be underlined as they appear on the screen.

To exit from Underline mode, simultaneously press the ESC and 0 keys or press the return key. Leave a line of underlined text on the screen, then type the following command and press the return key.

```
FOR X=0 TO 500:FOR Y=0 TO 30:
POKE 54784,29:POKE54785,Y:
NEXT:NEXT
```

Register 29 contains the number of the line within each character set that the 8563 must access to achieve underlining. Try other values in the FOR Y=0 TO 30 loop to see how they speed up and slow down the line movement.

Now, try writing a short three- or four-line program and then enter the following command in Direct mode.

```
POKE 54784,8: POKE 54785,3
```

Next, press the shift and CLR keys and then list your program. If you look closely, you'll see that you have two screens. On the top screen, four pixel lines are used for each character. The remaining four pixel lines are used for the same characters on the bottom screen. Register 8 holds some information about interlace modes that the 8563 uses to produce its display. Use the run/stop and restore keys to return your screen to normal.

Video 1 and Video 2 illustrate two registers that control the horizontal and vertical locations of the screen display.

These experiments are useful in that they serve to illustrate how the address/status and data locations (54784 and 54785, respectively) are used to communicate with the 8563 video controller. The rest of this article deals with the machine language techniques to read from and write to the 80-column screen.

80-COLUMN SCREEN READING AND WRITING

Video 3 is a program that demonstrates how to manipulate characters on the 80-column screen. Run it; then try some of the following commands in 80-column mode.

First, clear the screen, then enter

```
SYS 3328,0,0,1
```

You should see an *A* in the upper-left corner of the screen. The demonstration program gets the row number, column number and screen code for the character from the three numbers fol-

lowing the SYS address. Next, clear the screen and enter

```
SYS 3328,1,10,2
```

This displays a *B* at row 1 (the second row), column 11. The program multiplies the row number by 80 and adds the column number to get the character-map address, then puts the appropriate values in registers 18 and 19. Clear the screen once again and enter

```
SYS 3331,0,0,0
```

This demonstrates the attribute registers by printing a sample text string at the row and column determined by the numbers following the SYS address. Leave the message on the screen and enter

```
SYS 3334,0,0,4
```

This routine places the value *4* in the attribute map for each of the characters in the text string. You should see the message change color, or, if you're using a monochrome monitor, you should see the brightness level change. In place of the *4*, try different values between 0 and 15. With the message still on the screen, try the following commands in Direct mode.

```
SYS 3334,0,0,4+128
```

This swaps character sets.

```
SYS 3334,0,0,4+64
```

This reverses the text.

```
SYS 3334,0,0,4+32
```

This underlines the text.

```
SYS 3334,0,0,4+16
```

This causes the text to flash. Notice that the flash rate of the cursor is the same as that of the text. With the text still flashing, type the following in Direct mode.

```
POKE 54784,24: POKE
54785,XOR(PEEK(54785),32)
```

This command alters bit 6 of register 24 in order to increase the flash rate.

You can even control the 80-column screen from C-64 mode. With flashing text on the screen, type NEW and press the return key. Then enter the one-line program "1 GO64", press the return key and run the program. Switching your monitor to Composite mode will show you the C-64 start-up screen. But switching back to RGB mode reveals the text still in place and flashing on the 80-column screen.

Now that you've switched to 64 mode from 80-column 128 mode, load Video 3, list line 180, remove the first REM statement and press the return key. Now run the program and try the earlier examples, using the same SYS numbers. Switch the monitor to RGB mode to see the result of each SYS command. The last example won't work as is, because of the XOR command.

To access the 80-column screen from C-64 mode, you have to turn on the C-128 in 80-

column mode. If you power up the computer as a C-64 by holding down the Commodore key or placing a C-64 cartridge in the expansion port, the 80-column screen will not work.

The last program, Video 4, is a demonstration of hi-res graphics on the 80-column screen. Run it in either 40- or 80-column 128 mode. Using the Basic 7.0 commands, the program creates a hi-res 40-column screen, then copies it onto the 80-column screen. Switch your monitor between Composite and RGB modes to see the results.

If you see a flickering vertical line on the right side of your screen, you have a newer model 128, and you must make the changes to lines 40, 100 and 1001 as noted in the REMS in those listing lines. R

Hi-Res Writer

By Bruce Jaeger

RUN It Right

C-64; C-128 (in C-64 mode)

Have you ever designed a particularly nice high-resolution graphics screen and wished you could print something on it? Like your name? Or perhaps you wrote a hi-res game for the C-64, with sprites, sound effects, and so on, and found there was no easy way to print something as simple as the player's score on the screen.

While Comal, Simons' Basic and the C-128's Basic 7.0 all have built-in provisions for printing characters on a hi-res screen, the C-64 doesn't. However, with the accompanying program, Hi-Res Writer, it's now remarkably easy to place text on your C-64 hi-res screen.

WRITING ON THE HI-RES SCREEN

A shape table (like the set of drawing instructions on the Apple) isn't available on the C-64, but a bit-pattern character-plotting method is. To place an *A* on your C-64's hi-res screen, you Poke into screen memory the exact eight bytes of information that resemble the letter *A*.

Besides having a hi-res addressing arrangement that's fairly easy to deal with, the C-64 has another advantage in displaying text—you don't have to create and load a character set. Why should you, when the 64's own 4K character set ROM is available to borrow from?

By disabling the interrupts and changing the 6510's bank-switching pointers, you can look into the C-64's normally hidden character ROMs, located at $D000 (53248 decimal). At this address, you find both the upper/lowercase and uppercase/graphics sets, including all reversed characters. The next step is to discover how to find an individual character in this ROM and decide where and how to place it on the screen.

Each character is eight bytes long and is stored in the character ROM in the same alphabetical order that you might observe by Poking all the screen

codes onto the normal screen. The following demonstration program prints the alphanumeric characters, which are found in location 53272.

```
10 PRINT CHR$(147) : REM CLEAR
   SCREEN
20 POKE 53280,0 : POKE 53281,0 :
   REM BLACK BORDER & SCREEN
30 POKE 53272,21 : REM USE
   UPPERCASE CHARACTER SET
40 FOR X = 0 TO 255
50 POKE 1024+X, X : REM POKE
   CHARACTER ONTO SCREEN
60 POKE 55296+X,1 : REM POKE
   WHITE BEHIND IT IN COLOR RAM
70 NEXT X
80 GOTO 80
```

The first character set starts with "@ABCDEF...". To find, for example, the uppercase *B*, you take its ASC(x) value, which is 66, subtract 64 (getting 2), and then multiply by 8 (because each character uses eight bytes). This means that, since the uppercase/graphics set begins at 53248, the eight numbers defining the letter *B* begin at 53264 (53248 + 16). (The @ character is at 53248 to 53255; the *A* is at 53256 to 53263.) You always have to do a bit of trial-and-error work to get the proper character using its ASCII code.

Hi-Res Writer uses what are sometimes called screen Poke codes. The message is printed at the top of the normal text screen, which is not visible when you are in High-resolution mode. The English pound sign (£) follows the message to mark its end for the machine language routine.

When called, the machine language routine looks at the actual values in the memory addresses "behind" the letters (starting at 1024 decimal). This little trickery yields numbers that can be directly looked up in the character ROMs, saving lots of complicated (and slow!) mathematical computation.

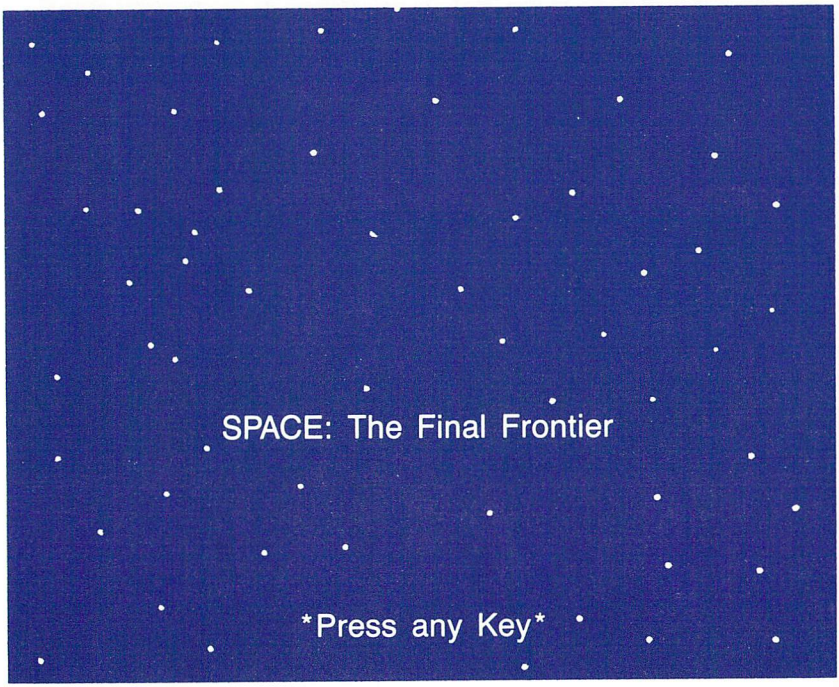
Of course, you needn't fully understand any of this to use Hi-Res Writer in your own programs. It's extremely easy to use.

Hi-Res Writer

Before running Hi-Res Writer, remove or disable any Basic-extension or cartridge programs; if you don't, your computer may lock up.

The High-Res Writer program first sets up the high-resolution bit map and clears the screen. It then sets all colors to a yellow foreground on a dark-gray background. A familiar quotation is printed on the screen, along with the prompt, "Press space bar to continue."

Take a look at the colors before pressing that space bar. Note that the words displayed have foreground and background colors that differ entirely from the rest of the screen—an effect not possible in normal Text mode. (You can achieve some-



what the same effect with the 64's Extended Background Color mode, but you're limited to a few characters and a total of only four colors; here you can use all 16.)

The second demonstration screen begins after you press the space bar. The screen is again cleared, and the color memory is set up with a black background and white foreground. (Note that the program sets its own colors independently.) Several white one-pixel "stars" are then randomly Poked onto the screen, followed by sound effects and a message.

USING HI-RES WRITER IN YOUR PROGRAMS

Theoretically, the eight-byte character definition can be placed anywhere on the high-resolution screen. I've limited the positioning of the characters to the same 40 horizontal and 25 vertical positions of the standard screen. This eliminates a lot of slow Basic number crunching and really isn't that much of a limitation. (See pp. 122–127 of

The Commodore 64 Programmer's Reference Guide for a complete description of the high-resolution bit-map addressing.)

You may easily use the hi-res text subroutine in your programs, along with the other machine language aids. The routine is written to use one of the favorite memory locations for the high-resolution bit map and color memory: the bit map is at $6000 (24576 decimal) and the color memory is at $5C00 (23552 decimal). Note that these are the same addresses used by City Software's Doodle! drawing program, so you can load a Doodle!-produced screen into your program.

First, of course, your program has to Poke or somehow load the machine language program into memory. I located the machine language at $8000 (32768 decimal), which normally isn't used when the high-resolution bit map is set at $6000.

Then, as in line 210 of the program, the top-of-Basic must be reset to protect the bit map (and the machine language code) from Basic variables.

Here's an explanation of how to set up and use the routine within Hi-Res Writer. Each time before calling the text subroutine, you have to make sure that the following five variables are set:

1. The message to be printed is stored in variable S$. Your message may contain any combination of alphanumeric characters; just don't use control characters, such as color changes or reverse.

2. The foreground color must be specified in variable FC. Use the standard values 0–15 for the colors black through light gray, and store the chosen value in variable FC. For example, if I want my message to be printed in light gray, I'd assign the value of 15 to FC before calling the text subroutine.

3. The background color is assigned in variable BC. Again, use the standard values 0–15 for the colors black through light gray, and store the chosen value in variable BC. For example: for a black background, let BC = 0.

4. The horizontal starting position of your message is stored in variable H. This is a value from 1 to 40.

5. Store the vertical starting position of your message (1–25) in variable V. For example, to start your message halfway down the left-hand side of the screen, assign H = 1 and V = 13.

After setting those five variables, just enter the Gosub command with the starting line number, and you'll be all set. R

Needlegraph

By Doug Venator

RUN It Right

C-64; C-128 (in C-64 mode); disk drive; printer

My wife recently asked me if I could write a program to help her with her latest needlepoint project. Naturally, I jumped at the chance to show what the computer could do.

She was creating colorful designs by weaving the yarn into little mesh squares, which she then sewed into cubes to form durable blocks for children to play with. She wanted a program that would simulate the mesh grid she was using. Her intention was to place vari-colored squares, representing the yarn stitches, on the grid, and then "edit" the colors until she got the design she wanted.

She also said it would be nice if she could make a printout of the design to save for later use. I set out to accommodate her, and the accompanying program, Needlegraph, seemed to fit the bill perfectly, with even a little bonus: The printouts formed useful little color-by-number designs that our kids found irresistible.

Program Description

The program, written in Basic, is fairly fast, although when it's compiled, the routines will be executed even more quickly. Once you run the program, the screen displays a 15 by 15 drawing grid of squares, represented by asterisks, along with instructions for most of the key commands.

The program begins in the Draw mode. In the upper-left corner is a blinking cursor, which you can move around the grid with the cursor controls in the same manner as in normal Basic. If you move it off any edge of the grid, it will wrap around to the opposite side.

Pressing the space bar will fill the cursor position with the color indicated by the square in the upper-left corner of the screen. To color a line of squares in the current color, hold down the space bar; the line will extend in the direction of the last cursor key pressed.

KEY COMMANDS

Number keys 1 to 4: In the upper-left corner of the screen, there is always a palette of four colors, indicated by three small circles and a square, the latter indicating the current color. (When black is one of the four on-screen colors, its circle—or square—is, of course, invisible against the black background.) You may make any of the four the current color by pressing the appropriate number key (1–4).

Function keys F1, F3, F5 and F7: These four keys correspond to the four on-screen colors. Pressing any one of these will increment the associated color circle (or square, for the current color) through all 16 of the 64's colors. In this way, you can make any combination of four colors quickly available, and all 16 can be used in the same design by simply changing one or more of the four you originally chose.

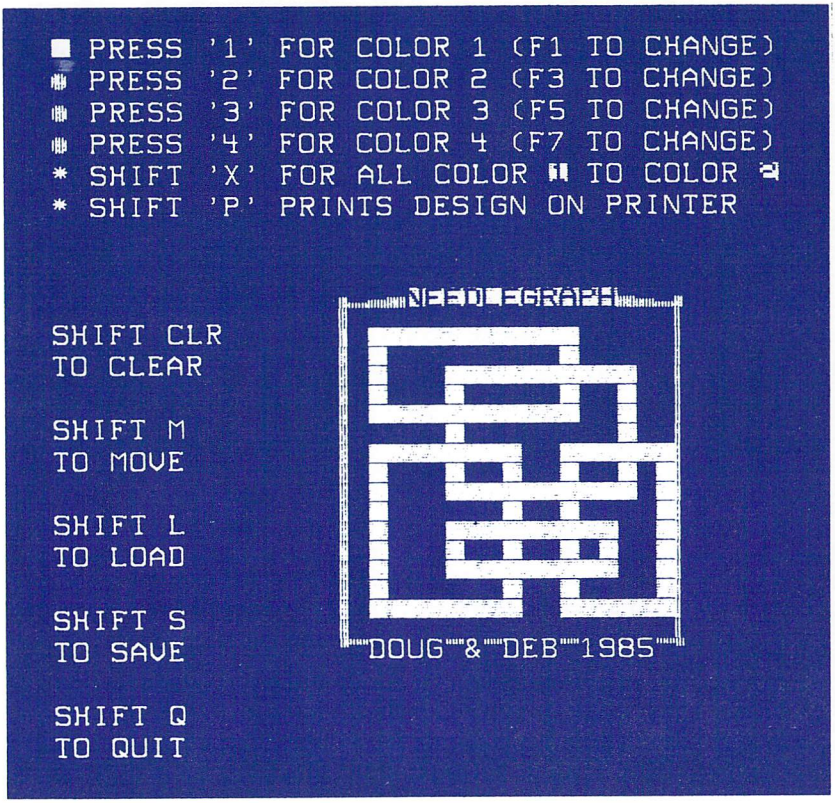
Shifted S: By simultaneously pressing the S and shift keys, you can save the displayed design to disk. The cursor stops blinking, and you are prompted for a filename, which can be any legal, disk drive-acceptable string. It's limited to 14 characters, however, since the program reserves the first two for prefixing a file identifier to the filename. This identifier aids in searching for and loading the saved designs, which are stored in sequential files.

If you type in a filename that already exists on the disk present in the drive, you'll be asked if you wish to replace the existing design on file with the one on the screen. If you type Y, the old file is scratched and replaced with the new one. If any other key is pressed, the program switches back to Draw mode, and you can save the design under another name by again using the shifted S.

Shifted L: You use this key combination to load a previously saved design from disk onto the drawing grid. Again, the cursor stops blinking, and you're prompted for a filename. If you're not sure what to call your design, you can type in a dollar sign for the filename, and the saved designs will be displayed on the right side of the screen, one disk block at a time.

You'll then be prompted to press any key to advance to the next set of filenames, until all have been displayed. At this time, the program will return to Draw mode, and you can again use the shifted L to load the desired design.

Shifted M: This key combination places the program in the Move mode. Now, if you press any cursor key, the displayed de-



sign will move one space in the indicated direction. Color squares shifted off the design grid will be rotated onto the opposite edge. In this way, you can move a design off one side of the grid and back onto the other.

You may move a design in any direction at any time, but I recommend that you first color in all the squares you want filled, since only colored squares are shifted.

An alternative to filling in the squares individually is to load a previously saved solid background before starting your design.

Shifted X: This combination enables you to change all squares containing color 1 to color 2.

Shifted P: This combination of keys prompts you for a title for the on-screen design, and then

prints the title, the design and a color key table to the printer. The design is printed as a grid of letters, with each letter corresponding to one of the 64's 16 colors as indicated in the color key table printed beside the grid. Each letter represents one stitch of yarn of the color indicated. As mentioned before, these printouts make great color-by-letter exercises that children greatly enjoy.

CLR/HOME: This key (unshifted) returns the blinking cursor to the upper-left corner of the drawing grid.

Shifted/CLR: This key combination clears the drawing grid of all colors and covers it with white asterisks, as if the program had just been loaded and run. The cursor will also return to the upper-left corner of the drawing grid. Be careful of this feature, for a lot of work can be erased if it's used at the wrong time.

Shifted Q: This combination restores use of the run/stop and restore keys, ends the program and returns the computer to Basic.

When I wrote this program, I certainly had no intention of publishing it, but several of my friends saw what it could do, asked for copies for their wives and urged me to submit it for publication. Apparently, there are a lot of people doing needlepoint or other similar design work who might find Needlegraph useful. I hope that many of you do. R

Extra! Extra! Read All About It

By David A. Hook

RUN It Right

C-64; C-128 (in C-64 mode)

The world of literature changed dramatically with the development of the printing press; no longer was manual transcription the only way to produce copies of valuable works. In our century, the photocopy machine proved to be another liberating step for mass access to the printed word. Yet these processes, whether manual or automatic, still involve traditional paper copies.

You have all heard of the "paperless society" that computerization is making possible. I invite you to take another step towards this goal.

ELECTRONIC NEWSLETTER

One of the benefits of our thriving 125-member local computer club, the Barrie User's Group (BUG), has been a free, 24-page monthly newsletter—an ambitious undertaking for a small group. Although we enjoyed the low-cost services of an "after-hours" printer, it became physically and financially impossible to produce and distribute so many printed pages.

A solution came to mind during the Christmas 1984 season. Robert Sims, whose telecommunications talents are now familiar to *RUN* readers, had started a disk-based magazine called *Bozart*. Program updates, technical information and a serialized work of fiction marked the early copies. Why, I wondered, couldn't our user's group put out a disk-based newsletter?

Inspired by *Bozart*, I began developing the process. Our newsletter text could be produced with the word processor, then put into the form of a text file. A file-reading program would then enable members to see the issue on their monitor screens.

Members were already turning in a disk each month to get our public domain library releases; now they could supply a second disk for the newsletter, and our problems would be solved. No commercial printer to pay, no lead time and virtually no cost! (We do have a number of members with dual disk drives, and no shortage of assistance in making up the disk copies.)

We quickly developed programs to produce the text file, read it to the screen and print a paper copy. Refinements have been made over the past year to include a colorful cover page (either KoalaPainter or Doodle format). Though we have been using the PaperClip word processor, the arrival of Robert Rockefeller's RUN Script 64 word processor (see *RUN*, March and April 1986) makes this system available to you for free.

Of course, a club newsletter is not the only use for this system. With it, you could provide documentation for another program in screen-readable, rather than printed, form. Maybe your family Christmas letter to computer-equipped relatives could take a revolutionary form.

After more than a dozen issues, our members are still enthusiastic. Lineups for the cover art are long—a lot of people have the graphics packages and love to share their creations. And I think we are less intimidating to potential writers in this form. The end product provides a bigger and better service to our membership.

Perhaps your local group has been unable to produce a newsletter for lack of funds or resources. If so, you might take a close look at our experience. It could be the answer for you, too.

REQUIREMENTS

Changing from a printed to an electronic newsletter requires some preparation. First, you must have a C-64, a disk drive and a copy of the RUN Script word processor.

The Koala Painter or Doodle cover page is optional. As the publisher, you don't need the graphics software. The finished picture may be prepared by anyone with these packages; then it can be copied over to the newsletter disk.

Use a file copy utility, such as Jim Butterfield's Unicopy, to transfer the Koala or Doodle picture to the newsletter disk. (Make sure its name matches the one you use in the Mag Reader Data statement, which will be explained later in this article.)

Each article or news item appearing in the newsletter must be prepared using RUN Script. You must add a few commands to create the special text files you

need later. These will be explained fully below.

Three programs make up the system: Mag Production, Mag Reader and Mag Printer. They're all written in Basic, so just follow the normal loading procedure. Be sure to save them, but don't run them yet, as we're not ready to use them.

Mag Production has all the functions necessary to prepare the large relative file (the entire newsletter). Mag Reader is a flexible, screen-oriented display program, which reads the relative file and sends it to the screen, one page at a time. Function keys control the various modes of operation as you step through the magazine. Mag Printer is for those who wish a partial or complete copy of the newsletter on paper.

Only the newsletter's editor uses Mag Production. The other two programs, Mag Reader and Mag Printer, must be copied to the newsletter disk for each issue, so that it can be read and copied.

THE EDITING PROCESS

Before using any of the three programs, contributors will have to prepare their articles on a word processor. You can accept article submissions in RUN Script or other compatible format (PaperClip sequential text files are usable). Tell your writers not to insert special codes, such as double spacing, justification, underlining, headers or page numbering; otherwise, you'll just have to remove them. Just have them send you the plain-Jane, raw-text file.

Exercise your editorial blue pencil via word processor until the article is ready for publication. Then, from within RUN Script, insert the following format line as the first line of the article (don't forget to terminate the line with a Return):

```
.pl25.pw41.lm1.rm1.tm1.bm1
```

This sets up a "screen page" of 25 lines, with minimum margins on all sides. At this point, you can save files as either program or sequential. Later, however, any program files will be converted to sequential form. Set aside two disks for your magazine preparation: a source disk and a newsletter disk. Save the article at this point, using appropriate (and short) filenames. Repeat this process for all the other articles in the issue. Your source disk should have text files only.

The newsletter disk will hold the articles from your source disk that you will modify slightly. (You should put the Mag Reader and Mag Printer on this disk, too.) Using RUN Script, you will transfer articles from the source disk to the newsletter disk. It will receive the modified text files, too.

Output from RUN Script will now be directed to the disk drive, instead of to a printer. Here's how to accomplish this. Reload the first article into RUN Script from the source disk. Insert the newsletter disk and activate the Print command. Specify "8" as the output device. You'll then be asked for a filename. If "text1" is the title of the first article, then respond with "p/text1,s,w". The output of text to the disk will be continuous, until the entire article has been stored. (Output it as a regular Commodore file, not straight ASCII.)

The prefix "p/" is a handy designation for these special "printer file" article conversions. The suffix ",s,w" specifies a sequential file. This is *not* optional.

Repeat the process: Load each article from its source disk, change disks and print it to the newsletter disk. Make note of the exact filenames. These "p/" files may be prepared in any order, not necessarily the order in which they will appear in the magazine.

Exit RUN Script for the time being. Now you are ready to use the first program contained in this article.

MAG PRODUCTION

Load Mag Production and list lines 7000–8000. The Data statements include a default issue name for the magazine, the number of the drive (usually 0) where the newsletter disk resides and a suggested length of 60 screen pages. Change these to suit your own preferences or equipment, then run the program. The menu provides six options.

1. Create. Select this option to create a new issue.

Insert the newsletter disk. The issue name (from the Data statement) will appear to the right of the question mark. You may change it if you wish; if you're satisfied, just press the return key.

The drive number and its default will be offered next, so reply accordingly. The third question offers a change in length from the default value.

By first changing the data values, you need only press the return key in response to each prompt, thereby saving time. It will take a few moments to create the relative file. When this is done, press the space bar to return to the menu.

2. Add. Select this option to add an article to the issue.

Respond to the first two prompts as before. The third prompt will ask for a text filename. You must assemble your news items in the order in which you want them read. Provide the filename of the first "p/" file (the lead news item of the issue). In the example above, you would

```
SOFTWARE MAGAZINE
(c)1986 David A. Hook

1. Create New Issue
2. Add To An Issue
3. Table Of Contents
4. Delete Text Files
5. Fix Page Counter
6. Quit Program

Choice (1-5): ?
```

respond "p/text1". Do *not* add the ",s,w" suffix! You'll have to enter the drive number next; then the "Working..." message will appear.

While the text file is being transferred to the relative file, the disk error light will blink on and off. This is normal, so you shouldn't be concerned. If there is a real disk error, the program will abort, giving a full description of the error. (Go have a cup of coffee if your text file is long; this process takes a while.)

Upon successful completion of the transfer, the starting and ending screen pages will be given to you. Record these on a sheet of paper, so that you can prepare the table of contents later. Note that p. 2 is the lowest page available, as p. 1 is reserved for the table of contents.

Continue accessing menu option 2 for each article to be added to the issue.

3. Table of Contents. With your handwritten record of article titles and their page numbers, you're ready to exit the Mag Production program and reload RUN Script. You will create the table of contents (TOC) using RUN Script and then print the file to disk as described in "The Editing Process," above.

I've provided a sample screen dump for your guidance (see Figure 1). You cannot exceed 22 lines of text or you'll clobber p. 2 of the issue. I suggest you design

your own model and save it for future issues. Once you get it right, you won't need to worry about overflow.

You must make the TOC into a "p/" file, as you did with each article. Print it to the newsletter disk. Exit RUN Script and reload Mag Production.

Select option 3 to create the table of contents. At the "contents" prompt, supply the "p/" filename you gave to the TOC, and it will be quickly transferred to the relative file.

4. Delete Text Files. This option has been provided for your convenience. The "p/" files are just the intermediate files and really don't belong on the finished newsletter disk. If you were consistent in using a prefix like "p/" for all of these, then option 4 would automatically delete them from the newsletter disk. Otherwise, you'll have to do this manually.

The prompt lets you use another prefix, in lieu of "p/". You'll have one last chance to abort before any scratching takes place.

This will most likely be the last step you'll perform on this newsletter, as it is now finished.

5. Fix Page Counter. This option might be called the "Oops" function. It lets you reset the ending page number to any value.

It may be that you inadvertently processed the files in the wrong order and would like to rebuild the magazine. You could always start from the beginning (Create), but that's often a lot of work. Use this option to set the page to the end of the last "good" news item, then resume the Add process.

Perhaps you've discovered a significant error in the finished issue? Make a (corrected) new "p/" file with RUN Script. Load and run Mag Production and note the current ending page. Reset the page counter just before the starting point of the old file, perform the Add, then reset again to the proper ending page.

You may find other uses for this option when something goes wrong. When finished with Mag Production, press 6 from the menu to exit the program. Next, you'll work on Mag Reader.

CUSTOMIZING MAG READER

You will have to customize Mag Reader for your own use. Load the program and list lines 6999–8000. The data items identify the variables that must be changed for each issue.

I chose a white background and border with light-blue text for easy readability. You can change the values in line 7000 to suit your preference.

The graphics picture information appears in line 7100. If it's a

Koala type, use its exact name, without the graphics symbol or the extra spaces at the end. The second field must be a "k" to signify its type. For a Doodle type, don't include the "dd" prefix in the field, and use "d" for the second field.

If you don't want to use a graphics cover, you must include a dummy name in the Data statement. Also, put an "n" in the second field, to indicate no graphics picture. The third field is the color for the border of the screen. Change this to suit your picture.

In line 7200, you need to provide the issue name, which is the exact name you chose when you created the issue with option 1 of Mag Production.

There is a title screen in the program. A date and a name for your newsletter should replace the data in line 7300.

Now you can save this modified version to the newsletter disk. I use a "month" suffix as a naming convention for the modified version (e.g., Mag Reader—jul). The user loads and runs this program to read the disk magazine.

CUSTOMIZING MAG PRINTER

Load Mag Printer and list lines 890–1000. Note the Data statement with the issue name. Correct this to the name you chose in the Create option. Save this modified version to the newsletter disk (Mag Printer—jul).

Now you've really completed the newsletter. Pass the disk over to your copying team to make duplicates for your members. You can take it easy until next month's deadline.

SOME TECHNICAL JAZZ

All three programs rely on a file-reading utility called The String Thing. This was originally composed for the PET/CBM computers by Bill MacLean, President of BMB Compuscience in Milton, Ontario. Jim Butterfield reworked the input function to make it more efficient. The Universal String Thing was published in the September/October 1982 issue of *The Transactor* magazine, and the follow-up C-64 version in that magazine's April 1983 issue.

The String Thing reads a file from disk without choking on commas, quotation marks or colons, as does Basic's Input# instruction. You define the input's maximum length in variable A$, the first variable used in the program (preset to length 40 or 41 here). A carriage return (in the file) will delimit the string. A RAM location holds the actual string length, and a Peek will fetch it. This machine language subroutine is a fantastic utility. The alternative to this would have been a convoluted (and slow) Get# loop.

The other machine language program appears in Mag Reader only. Its purpose is to move large portions of hi-res graphics characters into their proper place, then activate the correct graphics mode. RAM (location 2) will have been preset to identify the type of cover picture.

I've documented the programs with lots of REM statements, so you should be able to follow the logic fairly easily.

MAG READER FEATURES

You'll see a Help screen while the hi-res cover page loads from the disk. The graphics picture will then be displayed. Press any key or the space bar to advance beyond the cover picture.

Next, you'll see the text cover page, which gives your newsletter title and its issue month.

A prompt line will always appear on the bottom line of the screen. It identifies how to access the Help screen and indicates the page number you are reading. Since help is always available, the program is self-explanatory.

If you try to advance past the last page of the issue, the last page is redisplayed. If you back up from p. 1, you'll see the cover again. The table of contents is always just one keystroke away (F1); then you can go to any page using the F3 key.

MAG PRINTER FEATURES

This program will dump the issue either to the screen (device 3) or to the printer (device 4 or 5). Most printers that are interfaced to the C-64 will require that you set the secondary address to 7, so that upper- and lowercase will be printed.

Be sure that your printer is turned on. You must use continuous-form paper (fan-fold or roll). Advance the paper so that the perforation is one-half inch (three lines) above the print head. The program will print 60 lines of text, then skip over the perforation (six blank lines).

You may select the range of screen pages. If you wish to pause the printing, strike any key (except the run/stop). Resume by striking another key (Q will quit the program).

There you have it: a commercial-quality system that allows you to produce a software magazine with your C-64 and RUN Script word processor. After one issue, you'll find it to be a pretty routine process. It will neither tax your skill, nor take horrendous amounts of your time to produce the finished product. Then you'll be able to concentrate on the main function of an editor—flogging authors to get them to meet deadlines! R

Bootmaker 128/64

By Bill Pytlovany

RUN It Right

C-128; disk drive

One of the most refreshing features of the Commodore 128 is its ability to automatically load and execute specified C-128 programs that are on your disk. An example is the CP/M system disk packaged with the 128; you need only turn on the computer with this disk in the drive, and the CP/M operating system will be automatically activated. With some modification, you may apply this autoboot function to any disk used on the C-128.

Bootmaker 128/64 permits you to use this special feature in C-64 mode as well. And, with it, you no longer have to press down your Commodore key or type GO64 when powering up. The program works with the 1571, 1541 and other Commodore-compatible disk drives.

WHAT IT DOES

Bootmaker 128/64 performs two operations. First, it creates what is called a boot sector on track 1/sector 0 of your disk. If your disk drive is turned on when the C-128 off/on switch is activated or when the C-128 reset button is pushed, then this boot sector is checked for program information. Bootmaker 128/64 places in this sector the information needed to load a program called BOOT64.

Second, it performs the actual creation of the P1BOOT64 program on your disk. This machine language program contains the data required to switch the C-128 into C-64 mode, while still retaining the name of the program you wish to be loaded and executed.

HOW TO USE IT

Just load Bootmaker 128/64 into your C-128 computer in C-128 mode (in either 40- or 80-column) and run it. The program will instruct you to insert your C-64 disk and prompt you to enter the name of the C-64 program you wish to boot. After typing in the name of the pro-

gram and pressing the return key, the program will run for about 10–15 seconds while it displays the phrase "creating BOOT SECTOR" followed by "saving P1BOOT". It will then tell you when its operation has been completed, and you may test your newly bootable disk by pressing the reset button.

Sit back and watch. In a few seconds, your computer will switch to 64 mode. You will see the 64 start-up screen and a message from the author as your specified 64 program automatically loads and runs.

TIPS AND SUGGESTIONS

Try Bootmaker 128/64 first on a disk you can afford to lose—just as a precaution. You might take a newly formatted disk, put a copy of one of your C-64 programs on it and use it as a test.

Use extreme caution if you want to use the P1BOOT64 program on commercial programs. Many commercially produced disks may be damaged by writing any type of data to them. Make a working backup of your disk first!

Be sure you have a sufficient number of free blocks on your disk. If your disk is almost full, use a disk editor program to determine if track 1/sector 0 is free for use.

Use this program to autoboot a menu program such as those commonly available from on-line services or user group public domain libraries. (See *RUN*'s April issue for a C-64 menu program.)

So, instead of holding down the Commodore key or typing LOAD "whatever",8, get up out of your chair and s-t-r-e-t-c-h while your program loads. It really can't be beat! R

Disk Keeper

By Michael Broussard

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive

My computer workstation is usually stacked with piles of papers, books, disks and other paraphernalia. Cleaning up involves throwing away the obsolete listings, returning the books to their proper places on the bookshelf, putting the disks back into their jackets and filing them in their respective cabinets.

It was during one such cleaning binge that I wished I could carry the clean-up process down to the disk level. If only it were easier to make labels, get rid of obsolete files, protect important files from accidental deletion and perform other disk-keeping chores. Well, now I can, and you can, too—with Disk Keeper.

Disk Keeper is a menu-driven utility designed to help you easily manage your disks.

HOW TO USE THE PROGRAM

To use Disk Keeper, simply load the program and type RUN. Disk Keeper prints the following menu:

```
1) DISPLAY DIRECTORY
2) MAKE DISK LABEL
3) CLEAN UP DISK
4) PROTECT FILE
5) UNPROTECT FILE
6) RENAME DISK
7) FORMAT NEW DISK
8) SEND DISK COMMAND
9) EXIT
```

You are then prompted to choose one of the activities on the menu. Simply type in the number of your selection and press the return key. Depending on the option that you choose, you may be prompted for further information, such as a filename. When Disk Keeper is waiting for input, the cursor will be blinking in an input window. While the cursor is in this window, you'll notice that some keys (the cursor keys, for example) do not work.

To correct a typing error, you must use the DEL (delete) key to erase characters to the left of the cursor. Once you have typed the requested information into the window, press the return key. When an input window is empty, pressing the return key will usually re-

turn you to the main menu and abort the operation in progress.

Following is a description of what each menu selection does.

Display Directory. This menu selection displays the filenames, sorted in alphabetical order, on the disk currently in the drive. A typical entry might look like this:

```
7 MYPROG        PRG<
```

The sample entry shows that MYPROG is a program (PRG) file that takes up seven disk blocks. The "<" character after the filetype means that MYPROG is a protected file and can't be deleted. (Protected files are described in more detail later.)

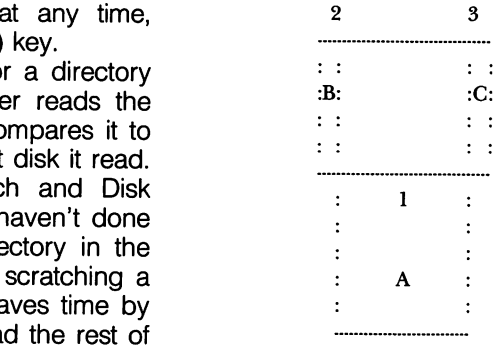
Ten file entries are shown at a time, and you advance to the next ten by pressing any key. (Press the cursor-up key to move back a screen.) When all the files have been displayed, pressing any key returns you to the main menu. To abort the display at any time, press the asterisk (*) key.

When you ask for a directory display, Disk Keeper reads the disk's name and compares it to the name of the last disk it read. If the names match and Disk Keeper thinks you haven't done anything to the directory in the meantime (such as scratching a file), the program saves time by not bothering to read the rest of the directory. It assumes that the file information it read the first time is still valid and uses it again.

Make Disk Label. Choosing this menu option allows you to make hard-copy listings of the information displayed by option 1. You are asked to choose a label format:

1) JACKET INSERT
2) FULL JACKET

If you choose format 1, the program sends to the printer an alphabetized listing that you can cut out and slip into the disk jacket along with the disk. If you have trouble keeping the label and the disk together, or if the disk jacket itself is becoming worn, you can choose format 2 instead. This produces a directory listing that you cut out and fold into a brand new disk jacket, with the filenames printed right on the front! Assembling the jacket is easy. Simply cut along the outside lines only, and fold as shown below.



Fold flap A at line 1. Fold flaps B and C at lines 2 and 3. Secure the flaps in the back with Scotch tape or glue. Warning: if you use

glue to secure the flaps, make absolutely sure the glue is dry before you put a disk in the jacket.

After specifying the output format, you are asked to supply an optional label string of up to 24 characters. Type in any characters you like, and they will appear on the listing next to the disk name.

If you have been reading *RUN* for a while, you may notice that the disk label output for the jacket insert is very similar to the output produced by my disk label utility published in the April 1984 issue ("Calling Disk Directories to Order"). There are a few differences, however.

The older program does not print the two-character disk ID. Also, Disk Keeper uses a completely different method of reading the directory, and the sort is now written in machine language and is *much* faster. For a disk with 45 files on it, Disk Keeper reads and sorts the directory three times faster than the previous version! In addition, Disk Keeper doesn't use any Commodore-specific graphics characters when printing labels, so it

is compatible with a wider range of printers.

Disk Keeper assumes that your printer is device #4 on the serial bus. If you are using a different device number for your printer, change the value assigned to the variable PR in line 110 of the program. (Don't forget to save your modified version.) Also, Disk Keeper assumes that your printer prints in standard pica type. If you have a printer that uses the smaller, elite typeface (such as Commodore's 1526), change from 0 to 1 the value assigned to the variable "Elite" in line 110. (Failure to do this will produce disk jackets that are too small if you choose format option 2.)

Disk Keeper prints its border and separator lines using the hyphen for horizontal lines and the colon for vertical lines. Depending on the graphics set of your printer, you might like to experiment by changing these to some other characters. This is easily done by changing the values assigned to the variables A$ and B$ in line 110. If you have a Commodore printer, try setting A$ = CHR$(192) and B$ = CHR$(221).

Clean Up Disk. After you have been using a disk for a while, it begins to fill up. A lot of the files eventually become obsolete. When you choose this menu option, Disk Keeper reads the disk directory. It then displays each filename in turn, asking if the file is to be deleted.

After you are asked about all the files on the disk, Disk Keeper goes through and deletes the ones you specified. Before the mass deletion starts, you are also asked if you want to validate the disk. If you answer Y, the disk is validated after the files have been scratched.

You can abort and return to the main menu without deleting any files by pressing the return key in response to any question before the deletion process actually begins. Also, once you have specified some files for deletion, you can skip the "Delete This File?" question for the rest of the files on the disk by typing in an asterisk in response to the prompt.

Protect File. (Caution: Do *not* use this feature with RUN Script 64 files.) A little-known feature of the disk operating system is the ability to protect a file so that it can't be accidentally deleted by the Scratch command. You need only type into the input window the name of the file you want to protect and press the return key. Protected files show up in the directory with a "<" next to the filetype.

Unprotect File. Once a file has been protected with the Protect File option above, it can't be

deleted (scratched) until it has been "unprotected." This option prompts you for the name of the file to unprotect, and turns the protect bit off in the directory.

Rename Disk. Although it is easy to change the name of a file using normal DOS commands, there is no regular DOS command for changing the name of a disk. When you choose this option, the name of the disk currently in the drive is displayed, and Disk Keeper prompts you for a new name. Type into the input window whatever new name you wish and press the return key. The disk will be renamed.

Format New Disk. Before a new disk can be used in the Commodore 1541 disk drive, it must be formatted. This involves assigning the disk a name and a two-character ID. (The disk ID is written onto every track and sector during the formatting process.) When you choose this menu option, Disk Keeper prompts you first for a name and then for a two-character ID for the new disk. You are warned that formatting the disk will destroy any existing files on it.

You are then asked if you wish to proceed; the disk will be formatted only if you answer Y to the question. (Note: When choosing a disk ID, be sure to select two characters you haven't used as an ID for any of your other disks. If DOS sees two disks with the same disk ID, it may mistakenly assume that the two disks are the same disk, with disastrous results to the files thereon.)

Send Disk Command. This option allows you to send to the disk drive any DOS command of your liking. You simply type the command into the input window and press the return key. Disk Keeper sends the command and then reads the error channel and displays the disk status. For example, to scratch a file named "OLD-PROG" from the disk, you would simply type:

```
S0:OLD-PROG
```

and press the return key.

Exit. Choosing this menu option allows you to exit from Disk Keeper and returns you to Basic.

Well, that's all there is to using Disk Keeper. R

Add Some Character to Your C-64

By Vito Solimene

RUN It Right

C-64; C-128 (in C-64 mode); printer optional

A nice feature of the Commodore 64 is its ability to let you redefine the current character set. With this ability, you can program any of the keyboard letters, numbers or graphics symbols to represent your own designs. You can then very easily place these new designs into your own programs.

However, to make a custom character set this way, you have to sit down with a pencil and some graph paper and draw every character you want to modify; you must then add up the bits to form the character. Once this is done, you have to take the data and Poke it into character memory to see how it looks.

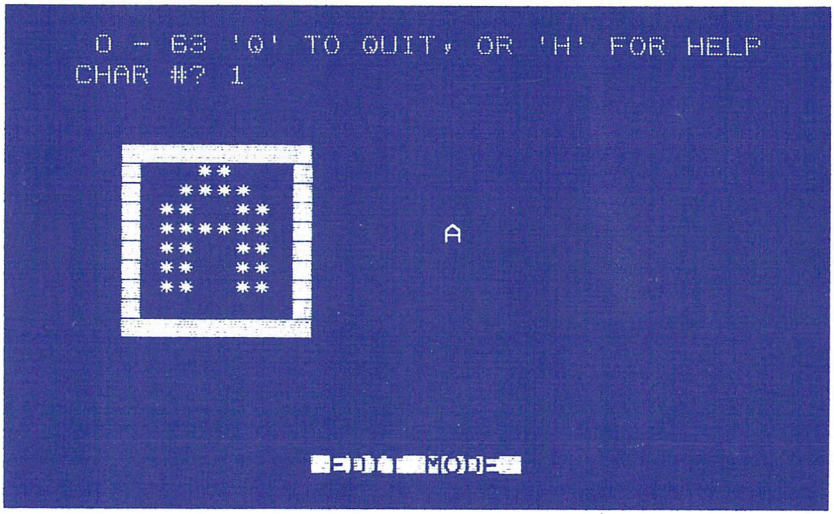
With my MAKECHARS program, you can easily create your custom character set without any tedious "pencil pushing" and immediately see the results.

THE PROGRAM

When you first run MAKECHARS, the screen will clear and display the message that the 64 is in the process of copying the character set.

The program copies the entire character ROM (all 4096 bytes) into RAM. When I first wrote the program, I used Basic to do the copying. However, it took too long, so I wrote the copy routine in machine language. The copy routine takes the characters at 53248 ($D000) and copies them into 12288 ($3000). (Keep in mind that rerunning the program will cause the character set to be recopied.)

After this is done, a prompt will



appear, asking you which character set you would like to edit. Answer by pressing the corresponding number. If you answer with an X, the program will end.

The next screen is known as the editing screen. This is where almost everything takes place. On the top of the screen, a message will appear ("XXX – YYY 'Q' TO QUIT 'H' FOR HELP CHAR#?"). The XXX is the lowest number allowed to be entered and the YYY is the highest. These numbers depend on the character set you picked earlier. This is known as the Prompt mode. A message at the bottom of the screen should appear to confirm this.

If you press H, a help screen will appear, giving a brief description of all the options available to you. If you enter Q, you'll be returned to the beginning of the program, where you can pick another character set to edit or quit the program.

Once you choose a number and press the return key, a box will be drawn, with the character you picked inside. The box is eight columns wide and eight lines high. Each line corresponds to one byte of the character shape, and each column corresponds to one bit in that byte. A few columns to the right of this box is the character you are currently working on, in its actual size. You are now in the Edit mode, and a message should

appear at the bottom of the screen confirming this.

In this mode, there are a lot of features available to you. If at any time you forget them, press H in the Prompt mode. The features are:

Cursor controls. The cursor-control keys enable you to move a flashing cursor around in the box.

Asterisk key (*). When you press this key, an asterisk is placed at the current cursor location. Note: This will not advance the cursor in any direction.

Space bar. When you press the space bar, a space is placed at the current cursor location. Like the asterisk, it does not advance the cursor in any direction.

CLR/HOME. Depressing this key doesn't change the character in any way; it just puts the cursor in the upper-left corner of the box.

Shift with CLR/HOME. If you simultaneously press the shift and CLR/HOME keys, the character box is cleared and your character erased.

CTRL 9. When you press the control key with the 9 key, the current character is reversed. Each asterisk becomes a space and each space becomes an asterisk.

N key. This advances you to the next character in numerical order without returning you to the Prompt mode.

P key. This is the exact opposite of the N command. It allows you to see the character before the one you are currently editing.

F1. Pressing this key while in the Edit mode causes the changes made in the enlarged character box to be transferred to memory and to be displayed. Note: Any alterations to the character will not be incorporated into the actual character until this key is pressed.

F3. This key will return you to the Prompt mode.

F5. Pressing this key allows you to save to disk or tape the character set you're working on. Once you press this key, a message will appear at the top of the screen, asking for the device to which you want to save your file. After you have answered, another prompt appears underneath, asking you for the file-name. If you pressed F5 by mistake, responding with an X returns you to the Prompt mode.

F6. Pressing this key allows you to load a character set from disk or tape. After you press F6, the screen clears, and a message appears, asking for the device from which to load your character file. The F6 feature has the same protection against accidental pressings as F5.

F7. Pressing this key allows you to print the data for the custom characters to screen or printer (you must instruct it by

entering either S or P). If, at any time, you wish to stop printing the data, press any key. To restart the printing, press C, for continue. To stop printing, press X, and you'll be returned to the Prompt mode. Be aware that the first number in the Data statements should not be used, since it is the number of the character and is for reference only.

Note: If you are working on the reverse uppercase character set, for example, and decide to save your file, it will be reloaded in the place of the normal reverse uppercase character set. This is true of each of the other character sets as well.

WHAT NOW?

So, you have a list of Data statements—now, what do you do with them? I've included another program, USECHARS, which takes care of this problem. I call it a skeleton program. It contains all you need in order to use the Data statements, except, of course, the data, which you will provide. R

The Choice is Yours

By Michael Broussard

RUN It Right

C-64

More and more programmers are using menu-driven programs for applications requiring a lot of interaction with the user. This article describes a subroutine you can use to make menu handling a snap.

Suppose, for example, that you have a program that expects as input the name of a color—blue, green or yellow. Part of the program might look like this:

```
10 INPUT "WHAT IS YOUR COLOR
   CHOICE";C$
20 IF C$ = "GREEN" THEN 100
30 IF C$ = "BLUE" THEN 200
40 IF C$ = "YELLOW" THEN 300
50 PRINT "INVALID COLOR" : GOTO 10
```

Although the above program fragment accomplishes what you want, it has a lot of disadvantages. For one thing, you have to type in the name of your color choice. If you misspell it, or type in a color the program is not expecting (red, for example), the program complains, and you have to type the color in again.

Another problem is that quite a few lines of Basic were needed to print the prompt and analyze the input. If you have to go through all that trouble every time you want to display a different menu, you are spending too much time programming.

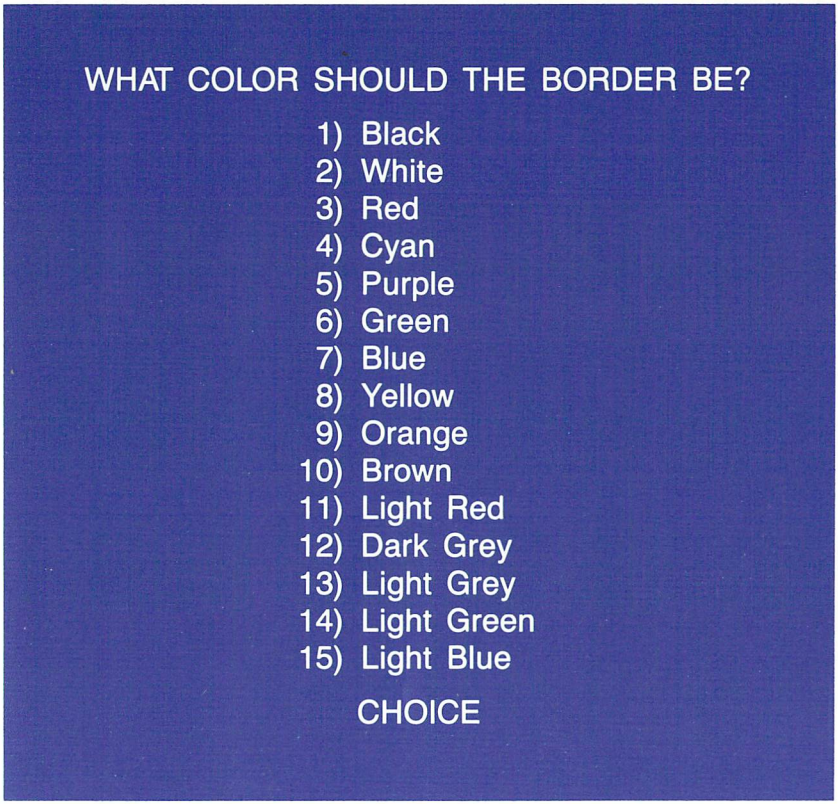
SUBROUTINE SHORTCUT

The solution is to use the subroutine described below. It prints a menu of numbered choices and then patiently waits until you pick one. A typical menu might look like this:

```
1) GREEN
2) BLUE
3) YELLOW
CHOICE?
```

Since there are three choices in this menu, the subroutine will print an error message if you try to input anything besides a 1, 2 or 3. After the error message is printed below the menu, the cursor moves back up to the CHOICE? prompt and gives you another chance. Once you make a valid selection, the subroutine returns to the main program.

The program demonstrates the



use of the menu subroutine. The program allows you to set the color of the monitor's border, background and characters by choosing the desired colors from menus.

Using the subroutine from your own programs is easy. All you have to do is set the variable P$ equal to the choices you want displayed, with each one preceded by the up arrow (↑) character. Then you set the variable PN to the number of leading spaces you want on the screen before the choices are printed.

The subroutine is invoked with a GOSUB statement and prints the menu in column PN + 1. It then inputs your choice. When the subroutine returns, the number representing the menu selection is stored in the variable P.

For instance, suppose you in-

clude the menu subroutine starting at line 500 in your program. To print the three-color menu used as an example earlier in this article, you could use the following program fragment:

```
50 P$="↑BLUE↑GREEN↑YELLOW":
   PN=15
60 GOSUB 500 : ON P GOTO
   100,200,300
```

Since the variable PN is set to 15, the menu will be printed in column 16 on the screen. (Actually, if there are less than 10 choices, the menu will start in column 17 when PN is set to 15. This is because an extra space is printed before choices numbered less than 10. Choices with numbers greater than 10 will be in column 16, so all the choices line up neatly.) When the subroutine returns to the main program, the variable P will be set to 1, 2 or 3, depending on which selection (blue, green or yellow) you made from the menu.

DEMO PROGRAM DETAILS

Let's consider the demonstration program for a moment, so you'll understand exactly what you must include in your programs in order to use the subroutine.

The lines in the program are numbered so that the ones *required* for inclusion in your program are multiples of 10. Any line number not ending in 0 is either a comment or is a part of the program not necessary for the subroutine. Of course, if you type in the program, you may omit the many REM statements.

In order to make the printing of the menu as fast as possible, part of the subroutine is coded in machine language. Lines 120 and 130 Poke the machine language instructions into memory. The instructions themselves are shown in the Data statements numbered 600 through 670.

The subroutine itself is lines 500 through 540. Those 14 lines (minus REMs) are all you need type in to use the subroutine. (In fact, once you're sure you've typed in the Data statements correctly, you can even delete line 130.)

Notice that P$ is the very first variable to appear in the program. It must be first, or the assembly language part of the subroutine will not work!

Also note that all the variables used in the subroutine have names starting with the letter P. By avoiding names of this form for the other variables in your program, you won't have to worry about the subroutine changing the value of one of your program variables unexpectedly.

The subroutine will handle up to 99 menu choices correctly, which should be plenty for most applications. (It is certainly more than will fit on a single screen!) Be careful in designing your programs that you don't give the

user too many choices in the same menu, for this tends to make the program harder to use.

In any event, try the subroutine. It will simplify your efforts and make your interactive programs more user-friendly. R

3-D Object Editor

By Louis R. Wallace

RUN It Right

C-128

Have you ever seen those graphics computer systems that allow you to create three-dimensional objects and rotate them in space? They are impressive and, for the most part, expensive. But with the proper software, your C-128 can also generate these 3-D objects.

If you've ever tried your hand at programming 3-D graphics, you're probably aware of the difficulties it entails. There's a lot of complex math involved just in converting an object defined in three-dimensional space to the plane of the computer screen. And if you want to rotate, scale or reflect the object, the mathematical overhead gets very heavy indeed. On top of everything else, you must have a way of entering into the computer the data that describes the object.

With the 3-D Object Editor, you can, in 40-column mode, create complex forms, manipulate them in various ways, store the new versions and even save the pictures you generate for use in other programs. The program contains both editing and manipulating features, which means you can use it to alter your data files, correct mistakes or make additions to line vectors you have already entered.

CREATING IN 3-D

Before you can manipulate or alter a 3-D object, you must first create it. The images are made from a series of 3-D line vectors. These are line segments defined by a pair of endpoints, but, unlike normal computer lines whose endpoints are each identified by two coordinates (X1,Y1; X2,Y2), these have an additional value, the Z coordinate (X1,Y1,Z1; X2,Y2,Z2).

As you know, on computer screens the X value refers to the horizontal axis, and the Y value to the vertical axis. The Z value refers to a third axis, which appears to come out of the screen toward you, giving the effect of three dimensions. The center of the screen is the origin, with co-

ordinates (0,0,0). To the right of the origin, the X axis is positive; to the left, negative. The Y axis is positive above the origin, and negative below. The Z axis is positive coming out of the screen toward you, and negative going into the screen away from you.

To create a 3-D object, you must have a clear idea of how you want it to look. The object must be definable as a series of points connected by lines, with the points having virtually any coordinate values you please, positive or negative. The first step in creating an object is to take a pencil and paper and draw it out, labeling each point with its appropriate X, Y and Z coordinates. Then decide which points must be connected by lines to give the object its form.

These lines, or vectors, are what you enter into the computer. For example, if you want to create a cube, you'd need to determine the eight endpoints, the vertices, that define its 12 edges. Starting out with one corner of the cube at the origin (0,0,0), and making each edge 10 units long, you can easily draw the set of 12 lines that describe the cube. Table 1 lists the coordinates for the endpoints of these lines. Figure 1 is a sketch of the finished cube.

To enter these lines, run the program and choose Input/Output from the main menu. This takes you to the second menu, from which you can type in data, get directories, read old files, save files or even save and restore screen images. To enter the cube data, select option 1, Enter New Data. This will give you the choice of reading a file from disk or entering data by hand.

Choose E (for Enter data), and you'll then be prompted to enter the pairs of coordinate triads for each line segment. After you enter the data for the last line (#12), the system will wait for a 13th line. By then entering an asterisk, you'll exit the data-entry option and be given a chance to review all, one or none of the line segments. If you think you've entered them correctly, press N for none; otherwise, review the data and make the necessary corrections.

Pressing N will return you to the main menu. You'd then be wise to choose option 1 (Input/Output) again, and, from *that* menu, option 5, which lets you save the object in memory to a disk file.

Once you've entered the cube and saved it to disk, you can begin looking at it. Option 1 on the main menu is Draw File, which translates the lines representing the 3-D cube in your file into an image that can be drawn on the two-dimensional screen. You are then asked several questions.

First, do you wish to have the

screen cleared before drawing? In most cases, you'd respond with Y for yes, but sometimes you might wish to have several versions of the same object on the screen simultaneously and so would not want the previous drawings erased.

Next, do you want to use Fast mode? The C-128 has a special feature that allows you to double the microprocessor speed so that your program runs about twice as fast. This is useful if there is some heavy computation involved, as there often will be in converting your data to screen coordinates. The drawback to Fast mode is that the screen will go blank while the computer calculates. If you have only a few lines (as with the cube), Fast mode is really unnecessary, but in the case of an object described with, say, 500 lines, it becomes a significant time-saver.

TRANSFORMATIONS

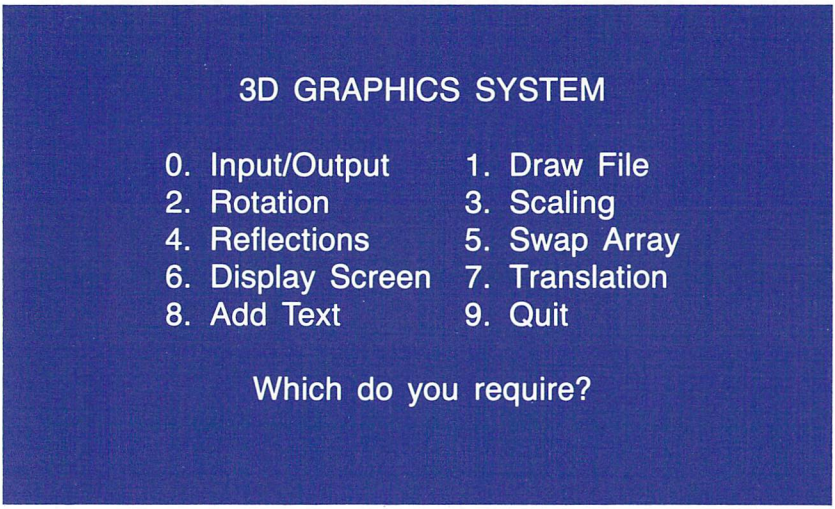
Next, the "viewing distance" is displayed on the screen, and you're asked if you want to change it. This value determines the apparent size of the object on the screen. The greater the distance value, the bigger the object appears. It is akin to shining a flashlight at a wall. The circle of light decreases in size as you move closer to the wall and increases as you move away. This changing of size is only visual; the data that describes the object is unchanged.

Once you decide about screen clearing, Fast mode and viewing distance, the computer will begin the mathematical calculations involved in creating the cube. In this case, it will take only a few seconds, and the cube will be drawn while you watch.

After you've seen the cube, you may want to transform it. Pressing the return key will bring you back from the view screen to the menu. For starters, I'd suggest choosing Rotation from the main menu, and rotating the cube 22 degrees about the Z axis. You'll be prompted for the information needed; just enter it at the proper places. Once the rotation is complete, you'll return again to the menu, where you can choose Draw File to see your newly rotated cube.

Another interesting transformation is to "scale" the 3-D figure, using option 3. Scaling simply means multiplying the values on each axis by some number, thus making the object bigger or smaller. A factor of 1 leaves the object unchanged, a fraction less than 1 reduces the size, and a number greater than 1 increases it. Also, each axis can be individually scaled, allowing you to distort the image in various directions.

If the image gets too large to



fit on the screen, nothing will be drawn, but the data can then be re-scaled down until the image fits. The system will not draw any line that is not completely on the screen. So if part of your image doesn't appear, you'll have to scale, rotate, reflect or translate it until it fits.

In addition to rotation and scaling, you can transform the object through reflection. This feature allows you to create a mirror image of the object with respect to one of three planes. For instance, if you choose to reflect along the plane determined by the Y and Z axes, all the X values of your object change sign. In other words, all positive X values (those to the right of the origin) become negative, while all negative X coordinate values, if any, become positive. Reflection along the X-Z and X-Y planes produces comparable mirror images.

Still another useful transformation feature is the ability to "translate" the object, which means to slide it in a given direction, keeping its orientation the same. One version of this simply moves the object to the origin, which is useful for centering it. But more versatility is possible with the regular Translate option, which allows you to move the object in any direction for any chosen distance. You just enter the amount (positive or negative) of the displacement, and the object is translated accordingly.

You are then returned to the main menu.

ALTERING AND SAVING

Since any of these transformations will change the data in memory, you should be sure to save the original file, as well as any altered versions you may want to use later. Perhaps you have a good image, but want to rotate it slightly to get a better view. You can temporarily save the current image—in case the transformed version turns out to be inferior—by choosing option 5, Swap Array. This will place your current data in a storage array. If you then want the original back again, return to Swap Array and recall it. But remember that this is not a permanent save; if the data is important, make sure it's stored on disk via the Save File option.

Since you'll very likely create pictures you want to show to others, there's a snapshot function on the Input/Output menu that's called Save Screen. This feature allows you to save the bit-map image as a 32-block binary file, which can then be reloaded for display with the Restore Screen option. You can also use these pictures in other programs, or print them out if you have the appropriate software.

Another feature is the ability to add to an array in memory. Let's say the cube is in memory, you also have on disk a pyramid shape, and you want to combine the files. From the main menu, choose 0, Input/Output. Then choose option 2, Alter Data. This is similar to the Enter New Data option, except that it appends the new data to that which is already in memory. With this, you can create complex images from simple ones. Again, you have the option to edit one, all or none of the lines after entering the new ones.

VISUALIZING IN 3-D

There are many applications for the 3-D Object Editor. It is, of course, very helpful in developing an understanding of three-dimensional spatial geometry. After a while, you'll find yourself becoming adept at thinking and visualizing in space, a skill that can carry over to many other areas of knowledge. Since the program can handle up to 500 lines, highly complex objects can be created, manipulated and stored.

By using any hi-res standard screen dump, you can get hard copies of your screens for illustrations in reports or other papers. You can also create pictures for use in other C-128 programs. But most of all, it's a lot of fun, and you'll find yourself spending a lot of time rotating, translating and scaling your designs. R

Please send me back issues of ReRUN

____ Spring Edition	____ C-64	____ Cassette version(s) at $11.47*
____ Gamepak	____ VIC-20	____ Disk version(s) at $21.47
____ Summer Edition		
____ Fall Edition		
____ Productivity Pak (Disk only)		
____ Winter Edition		
____ Jan./Feb. 1986 (Disk only)**		
____ March/April 1986 (Disk only)**		
____ May/June 1986 (Disk only)**		

** Prices include postage and handling. Foreign Air Mail please add U.S. $1.50 per item and $25 per subscription. Prepayment only.*
*** Includes 128 mode programs.*

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • Elm Street • Peterborough, NH • 03458

BEAT THE RUSH!

Please send me:

☐ **1 year (6 issues) for $89.97**

☐ **September/October ReRUN disk for $21.47.***

***Available in October.**

Includes programs for C-64 and C-128 (in both 64 and 128 modes).

Price includes postage & handling. Foreign Air Mail please add U.S. $1.50 per item and $25 per subscription. Prepayment only.

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • Elm Street • Peterborough, NH • 03458

9 RUN *Programs Included on this Disk:*

Hi-Res/Low-Res Graphics ► Disk Utilities ► Programming Aids ► Desktop Publishing ► Applications

From the July RUN:

- Hi-Res Writer
- Needlegraph
- Extra! Extra! Read All About It
- Video Fantasia (128)

From the August RUN:

- Bootmaker 128/64 (128)
- Disk Keeper
- Add Some Character to Your C-64

PLUS Bonus Programs:

- Menu Subroutine
- 3D Object Editor (128)

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

ReRUN • Elm Street • Peterborough, NH 03458

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

The programs in ReRUN are taken directly from listings prepared to accompany articles in *RUN* magazine. They will not run under all system configurations. Use the RUN It Right information included with each article as your guide.

CW COMMUNICATIONS/PETERBOROUGH